Truthful Mechanisms with Implicit Payment Computation[*]

Moshe Babaioff
Microsoft Research
Mountain View, CA 94043
moshe@microsoft.com

Robert D. Kleinberg[†]
Computer Science Department
Cornell University
Ithaca, NY 14853
rdk@cs.cornell.edu

Aleksandrs Slivkins
Microsoft Research
Mountain View, CA 94043
slivkins@microsoft.com

April 2010

Abstract

It is widely believed that computing payments needed to induce truthful bidding is somehow harder than simply computing the allocation. We show that the opposite is true for single-parameter domains: creating a randomized truthful mechanism is essentially as easy as a *single* call to a monotone allocation function. Our main result is a general procedure to take a monotone allocation rule and transform it (via a black-box reduction) into a randomized mechanism that is truthful in expectation and individually rational for every realization. Moreover, the mechanism implements the same outcome as the original allocation rule with probability arbitrarily close to 1, and requires evaluating that allocation rule only once.

Because our reduction is simple, versatile, and general, it has many applications to mechanism design problems in which re-evaluating the allocation function is either burdensome or informationally impossible. Applying our result to the multi-armed bandit problem, we obtain truthful randomized mechanisms whose regret matches the information-theoretic lower bound up to logarithmic factors, even though prior work showed this is impossible for truthful deterministic mechanisms. We also present applications to offline mechanism design, showing that randomization can circumvent a communication complexity lower bound for deterministic payments computation, and that it can also be used to create truthful shortest path auctions that approximate the welfare of the VCG allocation arbitrarily well, while having the same running time complexity as Dijkstra's algorithm.

ACM Categories and Subject Descriptors: J.4 [Social and Behavioral Sciences]: Economics; K.4.4 [Computers and Society]: Electronic Commerce; F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

Terms: theory, algorithms, economics.

Keywords: single-parameter mechanism design, truthful auctions, multi-armed bandits.

[*]This is the full version of the paper in *11th ACM Conf. on Electronic Commerce (EC)*, 2010.

[†]Parts of this research have been done while R. Kleinberg was a Consulting Researcher at Microsoft Research. He was also supported by NSF Awards CCF-0643934 and AF-0910940, an Alfred P. Sloan Foundation Fellowship, and a Microsoft Research New Faculty Fellowship.

1 Introduction

Algorithmic Mechanism Design studies the problem of implementing the designer's goal under computational constraints. Multiple hurdles stand in the way for such implementation. Computing the desired outcome might be hard (as in the case of combinatorial auctions) or truthful payments implementing the goal might not exist (as when exactly minimizing the make-span in machine scheduling [2]). Even when payments that will generate the right incentives do exist, finding such payments might be computationally costly or impossible due to online constraints.

It is widely believed that computing payments needed to induce truthful bidding is somehow harder than simply computing the allocation. For example, the formula for payments in a VCG mechanism involves recomputing the allocation with one agent removed in order to determine that agent's payment; this seemingly increases the required amount of computation by a factor of $n + 1$, where n is the number of agents. Likewise, for truthful single-parameter mechanisms the formula for payments of a given agent includes integrating the allocation function over this agent's bid. In some contexts with incomplete information, such as online pay-per-click auctions, computing these "counterfactual allocations" may actually be information-theoretically impossible. This calls into question the mechanism designer's ability to compute payments that make an allocation rule truthful, even when such payment functions are known to exist. Rigorous lower bounds based on these observations have been established for the communication complexity [7] and regret [8, 10] of truthful *deterministic* mechanisms.

In contrast to these negative results, we show that the opposite is true for single-parameter mechanisms that are truthful-in-expectation (over their random seed): computing the allocation and payments is essentially as easy as a *single* call to a monotone allocation function. This allows for positive results that circumvent the lower bounds for deterministic mechanisms cited earlier.

In more detail, our main result is a general procedure to take any monotone allocation rule[1] and transform it (via a black-box reduction) into a randomized mechanism that is truthful in expectation, universally ex-post individually rational,[2] implements the same outcome as the original allocation rule with probability arbitrarily close to 1, and requires evaluating that allocation rule only once. Further, if the original allocation rule is monotone in the ex-post sense, then the randomized mechanism is truthful in the ex-post sense as well. Our reduction applies in both the offline and online settings, and it applies to both Bayesian and dominant-strategy incentive compatibility.

Because our reduction is simple, versatile, and general, it has many applications to mechanism design problems in which re-evaluating the allocation function is either burdensome or informationally impossible. A leading problem for which only a single allocation can be evaluated is the multi-armed bandit (MAB) mechanism design problem [8, 10]. In this problem information about the state of the world is dynamically revealed during the allocation, so that the particular information that is revealed depends on the prior choices of the allocation, and in turn may impact the future choices. Simulating the allocation function on different inputs may therefore require information that was not revealed on the actual run. As per [8], this lack of information is a crucial obstacle for universally ex-post truthful MAB mechanisms: the appropriate payments cannot be computed unless the allocation rule is very "naive".

Applying our reduction to the MAB problem we derive that the problem of designing truthful MAB mechanisms reduces to the problem of designing monotone MAB allocation rules. In particular, using a monotone MAB allocation rule we obtain an MAB mechanism that is truthful in expectation, in the ex-

[1] An allocation rule is *monotone* if increasing one agent's bid while keeping all other bids the same does not decrease this agent's allocation.

[2] A mechanism is *individually rational* if an agent never loses by participating in the mechanism and bidding truthfully. Here and elsewhere in the paper, *universally* and *ex-post* refer to properties that hold for each realization of the random seed of the algorithm and nature, respectively.

post sense, universally ex-post individually rational, and has regret $O(T^{1/2})$ for the stochastic version of the problem. This regret bound matches the information-theoretic lower bound for stochastic multi-armed bandit problems that holds even in the absence of incentive constraints. This stands in contrast to the lower bound of [8], where it was shown that universally ex-post truthful mechanisms must suffer regret $\Omega(T^{2/3})$.

As a by-product of our results, we obtain an unconditional separation between the power of ex-post truthful-in-expectation and universally ex-post truthful mechanisms, in the online setting. This complements the recent result of Dobzinski and Dughmi [11], which gives a separation between these two classes of randomized mechanisms in the *offline* setting, under a polynomial communication complexity constraint. It is worth noting that the separation in [11] applies to a rather unnatural problem (two-player multi-unit auctions in which if at least one item is allocated, then all items are allocated and each player receives at least one item) whereas our separation result is for a natural problem (online pay-per-click ad auctions for a single slot, with unknown clickthrough rates).

Our main result also has implications for offline mechanism design. Nisan and Ronen, in their seminal paper on algorithmic mechanism design [15], cite the apparent n-fold computational overhead of computing VCG payments and pose the open question of whether payments can be computed faster than solving n versions of the original problem, e.g. for VCG path auctions. Our result shows that the answer is affirmative, if one adopts the truthful-in-expectation solution concept and tolerates a mechanism that outputs an outcome whose welfare is a $(1+\epsilon)$-approximation to that of the VCG allocation, for arbitrarily small $\epsilon > 0$. Babaioff et al. [7] present a social choice function f in an n-player single-parameter domain such that the deterministic communication complexity required for truthfully implementing f exceeds that required for evaluating f by a factor of n. Our result shows that no such lower bound holds when one considers randomized mechanisms, again allowing for a small amount of random error in the allocation.

Map of the paper. On a high-level, this paper makes three contributions. Our main result is the general reduction, as described above (Section 3). Then we have the two applications to off-line mechanism design (Section 4). Finally, we consider multi-armed bandit (MAB) mechanism design (Section 5). Here the main issue is designing monotone MAB allocations, which has not been studied in the rich literature on MAB problems. We make several contributions in that direction, including a general theorem stating that a number of existing MAB algorithms give rise to monotone MAB allocations (Section 5.2), and a new ex-post monotone algorithm for the stochastic setting (Section 5.3). As a by-product, we obtain an unconditional separation between the power of ex-post truthful-in-expectation and universally ex-post truthful mechanisms (Section 5.4). Presenting our results requires a significant amount of background material on algorithmic mechanisms design (Section 2) and multi-armed bandits (Section 5.1).

Related work. The characterization of truthful mechanisms for single-parameter agents, given by Myerson [14] for single-item auctions and by Archer and Tardos [2] for a more general class of single-parameter problems, states that a mechanism is truthful if and only if its allocation rule is monotone and its payment rule charges each agent its value for the realized outcome, minus a correction term expressed as an integral over all types lower than the agent's declared type. (See Eq. (4).) Exact computation of this "Myerson integral" may be intractable, but Archer, Papadimitriou, Talwar, and Tardos [1] developed a clever workaround: one can use random sampling to compute an unbiased estimator of the Myerson integral, at the cost of evaluating the allocation function once more. Thus, for n agents, the allocation function must be evaluated $n+1$ times: once to determine the actual allocation, and once more per agent to determine that agent's payment. Our reduction relies on a generalization of the same random sampling technique, but we show how to avoid recomputing the allocation function when determining each agent's payment.

As mentioned earlier, the question of whether the computational cost of computing payments is inherently greater than that of computing an allocation was raised by Nisan and Ronen [15] in the context of VCG path auctions. The most significant progress on this question to date was the communication complexity lower bound of Babaioff et al. [7], who constructed a single-parameter domain with a monotone social choice function (the so-called $\textsc{Not-Too-Far}_k$ function) whose communication complexity is n times less than the communication complexity of any deterministic incentive-compatible mechanism implementing $\textsc{Not-Too-Far}_k$. In contrast, our results show that no such lower bound arises when one considers randomized truthful-in-expectation mechanisms with arbitrarily small probability of outputting the wrong allocation.

The computation of payments in online mechanism design is a central issue in the analysis of truthful MAB mechanisms in [8, 10]. The analysis of such mechanisms is motivated by the problem of designing pay-per-click ad auctions when there is uncertainty about clickthrough rates. In the absence of incentive constraints, the problem of learning clickthrough rates can be modeled as a MAB problem, and it is known that there are algorithms whose regret (roughly speaking, the lose due to not knowing the clickthrough rates at the outset of the auction) is $O(\sqrt{T})$ where T is the number of impressions; moreover, this dependence on T is information-theoretically optimal.

The main result of [8] provides a characterization of deterministic mechanisms that are truthful for every possible realization of the other agents' bids and the users' clicks (the characterization extends to randomized mechanisms that are universally truthful). The characterization is more restrictive than the Myerson and Archer-Tardos characterization of truthful single-parameter mechanisms, because computing an agent's payment requires knowing how many clicks she would have received if she had submitted a lower bid value, and it is typically impossible to obtain this information in the online setting, since it is impossible to go back into the past and allocate impressions to a different bidder for the purpose of seeing whether a user would have clicked on that bidder's advertisement. Using the characterization of universally truthful MAB mechanisms, [8] proves that any such mechanism must incur regret $\Omega(T^{2/3})$. Here we show that randomized mechanisms, that are truthful-in-expectation and individually rational for every realization, universally ex post individually rational can achieve regret $O(T^{1/2})$, matching the information-theoretic lower bound for MAB problems in the absence of incentive constraints.

Finally, *dynamic auctions* [9, 3] is another setting in which information is revealed "dynamically" (over time). However, while in MAB auctions all information from the agents (the bids) is submitted only once and then information is revealed to the mechanism by nature over time, in dynamic auctions the agents continuously observe private "signals" from nature and submit "actions" to the mechanism. Accordingly, providing the right incentives becomes much more challenging. On the other hand, existing work [9, 3] has focused on a fully Bayesian setting with known priors on the signals, whereas all of our results do not rely on priors.

2 Preliminaries

Single Parameter Domains. We present the single parameter model for which we apply our procedure. The model is very similar to the model of Archer and Tardos [2], yet it is slightly more general. We state the model is terms of values and not costs and allow the values to be both positive and negative. We also allow randomization by nature. All these changes are minor and do not change the fundamental characterization, yet are helpful to later derive our results.

Let n be the number of agents and let $N = [n]$ be the set of agents. Each agent $i \in N$ has some private *type* consisting of a single parameter $t_i \in T_i$ that describes the agent, and is known only to i, everything else is public knowledge. We assume that the domain T_i is an open subset of $\Re$ which is an interval with positive length (possibly starting from $-\infty$ or going up to ∞). Let $T = T_1 \times T_2 \times ... \times T_n$ denote the domain of types and let $t \in T$ denote the vector of true types.

There is some set of *outcomes* O. For single-parameter domains, agents evaluate outcomes in a particular way that we describe next. For each agent $i \in N$ there is a function $a_i : O \to \Re_+$ specifying the *allocation to agent* i. The *value* of an outcome $o \in O$ for an agent $i \in N$ with type t_i is $t_i \cdot a_i(o)$. The *utility* that agent $i \in N$ derives from outcome $o \in O$ when he is charged p_i is quasi-linear: $u_i = t_i \cdot a_i(o) - p_i$.

For instance, consider the allocation of k identical units of good to agents with additive valuations: agent i has a value of t_i *per unit*. An outcome o specifies how many items each agent receives, that is, $a_i(o)$ is the number of items i receives. His valuation for that outcome is his value per-unit times the number of units he receives.

A (direct revelation) deterministic *mechanism* $\mathcal{M}$ consists of the pair $(\mathcal{A}, \mathcal{P})$, where $\mathcal{A} : T \to O$ is the *allocation rule* and $\mathcal{P} : T \to \Re^n$ is the *payment rule*, i.e. the vector of payment functions $\mathcal{P}_i : T \to \Re$ for each agent i. Each agent is required to report a type $b_i \in T_i$ to the mechanism, and b_i is called the *bid* of agent i. We denote the vector of bids by $b \in T$. The mechanism picks an outcome $\mathcal{A}(b)$ and charges agent i payment of $\mathcal{P}_i(b)$. The allocation for agent i when the bids are b is $\mathcal{A}_i(b) = a_i(\mathcal{A}(b))$ and he is charged $\mathcal{P}_i(b)$. Agent i's utility when the agents bid $b \in T$ and his type is $t_i \in T_i$ is

$$u_i(t_i, b) = t_i \cdot \mathcal{A}_i(b) - \mathcal{P}_i(b) \tag{1}$$

We also consider randomized mechanisms, which are distribution over deterministic mechanisms. For a randomized allocation rule $\mathcal{A}_i(b)$ and $\mathcal{P}_i(b)$ will denote the *expected* allocation and payment charged from agent i, when the bids are b. The expectation is taken over the randomness of the mechanism. Sometimes it will be helpful to explicitly consider the deterministic allocation and payment that is generated for specific random seed. in this case we use w to denote the random seed and use $\mathcal{A}_i(b; w)$ and $\mathcal{P}_i(b; w)$ to denote allocation and payment when the seed is w.

It is possible that there is some outside randomization that influences the outcome and is not controlled by the mechanism. We call this randomization by *nature*. An example for that would be the randomness in the realization of clicks in sponsored search auction. With such randomization $\mathcal{A}_i(b)$ and $\mathcal{P}_i(b)$ also encapsulate expectations over nature's randomization. Finally, we use the notation $\mathcal{A}_i(b; w, r)$ and $\mathcal{P}_i(b; w, r)$ to denote the allocation and payment charged from agent i, when the bids are b, the mechanism random seed is w and nature's random seed is r.

Allocation and Mechanism Properties. Let b_{-i} denote the vector of bids of all agents but agent i. We can now write the vector of bids as $b = (b_{-i}, b_i)$. Similar notation will be used for other vectors.

We next list two central properties, truthfulness and individual rationality.

- Mechanism $\mathcal{M}$ is *truthful* if for every agent i truthful bidding is a *dominant strategy* in $\mathcal{M}$. That is, for every agent i, bidding t_i always maximizes her utility, regardless of what the other agents bid. Formally,
$$t_i \cdot \mathcal{A}_i(b_{-i}, t_i) - \mathcal{P}_i(b_{-i}, t_i) \geq t_i \cdot \mathcal{A}_i(b) - \mathcal{P}_i(b) \tag{2}$$
holds for every agent $i \in N$, type $t_i \in T_i$, bids of others $b_{-i} \in T_{-i}$ and bid $b_i \in T_i$ of agent i.

- Mechanism $\mathcal{M}$ is *individually rational (IR)* if an agent never ends up with negative utility by participating in the mechanism and bidding truthfully. Formally,
$$t_i \cdot \mathcal{A}_i(b_{-i}, t_i) - \mathcal{P}_i(b_{-i}, t_i) \geq 0 \tag{3}$$
holds for every agent $i \in N$, type $t_i \in T_i$ and bids of others $b_{-i} \in T_{-i}$.

It will be helpful to establish terminology for the case that the above hold not only in expectation but also for specific realizations. For example, we will say that a mechanism is *universally truthful* if Eq. (2) holds not only in expectation over the mechanism's randomness, but rather for every realization of that randomness. In general, every property that we define is defined by some inequality, and if the inequality holds for every realization of the mechanism randomness we say that it holds *universally*, and if it holds for every realization of nature randomness we say that it holds *ex-post*. When we want to emphasize that the property holds only in expectation over the nature's randomness we say that it holds *stochastically*.

Note that in an individually rational mechanism an agent is ensured not to incur any loss *in expectation*. That is rather unsatisfying as for some realizations the agent might suffer a huge loss. It is more desirable to design mechanisms that are *universally ex-post individually rational*, that is a truthful agent should incur no loss for every bids of the others and *every realization* of the random events (not only in expectation).

If all types are positive, then in addition to individual rationality it is desirable that all agents are charged a non-negative amount; this is known as the *no-positive-transfers* property. Finally, the *welfare* of a truthful mechanism is defined to be the total utility $\sum_i t_i \cdot \mathcal{A}_i(t)$.

Characterization. The following characterization of truthful mechanisms, due to Archer and Tardos [2], is almost identical to the characterization presented by Myerson [14] for truthful mechanisms in the special case of single item auctions. The crucial property of an allocation that yields truthfulness is *monotonicity*, defined as follows:

Definition 2.1. *Allocation rule $\mathcal{A}$ is* monotone *if for every agent $i \in N$, bids $b_{-i} \in T_{-i}$ and two possible bids of i, $b_i \geq b_i^-$, we have $\mathcal{A}_i(b_{-i}, b_i) \geq \mathcal{A}_i(b_{-i}, b_i^-)$.*

Recall that monotonicity of an allocation rule is also defined universally and/or ex-post.

We next present the characterization of truthful mechanisms. In the theorem statement, the expression $\mathcal{A}_i(b_{-i}, u)$ is interpreted to equal zero when $u \notin T_i$.

Theorem 2.2. *[14, 2] Consider a single-parameter domain. An allocation rule $\mathcal{A}$ admits a payment rule $\mathcal{P}$ such that the mechanism $(\mathcal{A}, \mathcal{P})$ is truthful if and only if $\mathcal{A}$ is monotone and moreover for each agent i and bid vector b it holds that $\int_{-\infty}^{b_i} \mathcal{A}_i(b_{-i}, u)\, du < \infty$. In this case the payment for each agent i must satisfy*

$$\mathcal{P}_i(b) = \mathcal{P}_i^0(b_{-i}) + b_i\, \mathcal{A}_i(b_{-i}, b_i) - \int_{-\infty}^{b_i} \mathcal{A}_i(b_{-i}, u)\, du, \tag{4}$$

where $\mathcal{P}_i^0(b_{-i})$ does not depend on b_i.

A mechanism is called *normalized* if for each agent i and every bid vector b, zero allocation implies a zero payment: $\mathcal{A}_i(b) = 0 \Rightarrow \mathcal{P}_i(b) = 0$.

Corollary 2.3. *The truthful mechanism in Theorem 2.2 is normalized if and only if $\mathcal{P}_i^0(b_{-i}) \equiv 0$, in which case the mechanism is also individually rational and for positive-only types ($T \subset \Re_+^n$) it moreover satisfies the no-positive-transfers property.*

Remark 2.4. *Both Theorem 2.2 and Corollary 2.3 hold in the "ex-post" sense (resp., "universal" sense), if $\mathcal{A}_i$, $\mathcal{P}_i$ and $\mathcal{P}_i^0(b_{-i})$ are interpreted to mean the respective values for a specific random seed of nature (resp., mechanism).*

Note that for Corollary 2.3 to apply, truthfulness and normalization must hold in the same "sense", e.g. both ex-post.

3 The Generic Procedure

This section presents our procedure that takes any monotone allocation rule for a single-parameter domain and creates a randomized truthful-in-expectation mechanism that is ex-post individually rational and with identical outcome as the original allocation rule with high probability. The procedure uses the allocation rule as a "black box," calls it only once and allocates accordingly.

We begin with an informal description of our procedure before giving its formal definition. As evidenced by Eq. (4), the payment for agent i is a difference of two terms: the agent's reported utility (i.e., the product of her bid and her allocation), minus the integral of the allocation assigned to every smaller bid value. We charge the agent for her reported utility, and we give her a random rebate whose expectation equals the required integral. When integrating a function over a finite interval, an unbiased estimator of the integral can be obtained by sampling a uniformly random point of that interval and evaluating the function at the sampled point. This idea was applied, in the context of mechanism design, by Archer et al. [1]. Below, we show how to generalize the procedure to allow for integrals over unbounded intervals, as required by Eq. (4). Using this procedure it is easy to transform any monotone allocation rule into a randomized mechanism that is truthful in expectation and only evaluates the allocation function $n+1$ times: once to determine the actual allocation, and once more per agent to obtain an unbiased estimate of that agent's payment.

Our main innovation is a transformation that uses the same random sampling trick, but only needs to evaluate the allocation function once. Assume that a parameter $\mu \in (0,1)$ is given. For every player, with probability $1-\mu$, we leave their bid unchanged; with probability μ, we sample a smaller bid value. The allocation rule is invoked on these bids. An agent is always charged her reported value of the outcome, but if her bid was replaced with a smaller bid value then we refund her an amount equal to an unbiased estimator of the integral in Eq. (4), scaled by $1/\mu$ to counterbalance the fact that the refund is only being applied with probability μ. A naïve application of this plan suffers from the following defect: the random resampling of bids modifies the expected allocation vector, so we need to obtain an unbiased estimator of the integral of the *modified* allocation function. However, if we change our sampling procedure to obtain such an estimate, then this modifies the allocation function once again, so we will still be estimating the wrong integral! What we need is a "fixed point" of this process of redefining the sampling procedure. Below, we give a definition of *self-resampling procedures* that satisfy the requisite fixed point property, and we give two simple constructions of self-resampling procedures. A key feature of our definition is that a self-resampling procedure actually transforms a single bid into *two* correlated random values: one to be used in computing the allocation, the other to be used (together with the allocation itself) in computing payments.

Thus, the formal description of our procedure consists of three parts: **(i)** a method for estimating integrals by evaluating the integrand at a randomly sampled point, **(ii)** the definition and construction of self-resampling procedures, **(iii)** the generic transformation that uses the foregoing two ingredients to convert any monotone allocation rule into a truthful-in-expectation randomized mechanism. We now specify the details of each of these three parts.

3.1 Estimating integrals via random sampling

Let I be a nonempty open interval in $\Re$ (possibly with infinite endpoints) and let g be a function defined on I. Let us describe a procedure for estimating the integral $\int_I g(z)\,dz$ by evaluating g at a single randomly sampled point of I. The procedure is well known; we describe it here for the purpose of giving a self-contained exposition of our algorithm.

Theorem 3.1. *Let $F : I \to [0,1]$ be any strictly increasing function that is differentiable and satisfies $\inf_{z \in I} F(z) = 0,\ \sup_{z \in I} F(z) = 1$. If Y is a random variable with cumulative distribution function F, then the expected value of $g(Y)/F'(Y)$ is equal to $\int_I g(z)\,dz$.*

Proof. Since $\inf_{z \in I} F(z) = 0$ and $\sup_{z \in I} F(z) = 1$, it follows that the random variable Y is supported on the entire interval I. Our assumption that F is differentiable implies that Y has a probability density function, namely $F'(z)$. Thus, for any function h, the expectation of $h(Y)$ is given by $\int_I h(z)F'(z)\,dz$. Applying this formula to the function $h(z) = g(z)/F'(z)$ one obtains the theorem. $\square$

3.2 Self-resampling procedures

The basic ingredient of our generic transformation is a procedure for taking a bid b_i and a random seed w_i, and producing two random numbers $x_i(b_i; w_i)$, $y_i(b_i; w_i)$. The mechanism will use $\{x_i(b_i; w_i)\}_{i \in N}$ for determining the allocation and additionally $y_i(b_i; w_i)$ for determining the payment it charges agent i. To prove that the mechanism is truthful in expectation we will require the following properties.[3]

Definition 3.2. *Let I be a nonempty interval in $\Re$. A* **self-resampling procedure** *with support I and resampling probability $\mu \in (0,1)$ is a randomized algorithm with input $b_i \in I$, random seed w_i, and output $x_i(b_i; w_i)$, $y_i(b_i; w_i) \in I$, that satisfies the following properties:*

1. *For every fixed w_i, the functions $x_i(b_i; w_i)$, $y_i(b_i; w_i)$ are non-decreasing functions of b_i.*
2. *With probability $1 - \mu$, $x_i(b_i; w_i) = y_i(b_i; w_i) = b_i$. Otherwise $x_i(b_i; w_i) \le y_i(b_i; w_i) < b_i$.*
3. *The conditional distribution of $x_i(b_i; w_i)$, given that $y_i(b_i; w_i) = b'_i < b_i$, is the same as the unconditional distribution of $x_i(b'_i; w_i)$. In other words,*
$$\Pr[\, x_i(b_i; w_i) < a_i \mid y_i(b_i; w_i) = b'_i \,] = \Pr[\, x(b'_i; w_i) < a_i \,], \quad \forall a_i \le b'_i < b_i.$$
4. *Consider the two-variable function*
$$F(a_i, b_i) = \Pr[y_i(b_i; w_i) < a_i \mid y_i(b_i; w_i) < b_i],$$
which we will call the distribution function *of the self-resampling procedure. For each b_i, the function $F(\cdot, b_i)$ must be differentiable and strictly increasing on the interval $I \cap (-\infty, b_i)$.*

As it happens, it is easier to construct self-resampling procedures with support $\Re_+$, and one such construction that we call the *canonical self-resampling procedure* (Algorithm 1) forms the basis for our general construction. We defer further discussion of self-resampling until after we have described and analyzed the generic transformation.

3.3 The generic transformation

Suppose we are given a monotone allocation rule $\mathcal{A}$ and for each agent $i \in N$ a self-resampling procedure that has resampling probability $\mu \in (0,1)$, support T_i, and output values $f_i = (x_i, y_i)$. Let $F_i(a_i, b_i)$ denote the distribution function of the self-resampling procedure for agent i, and let $F'_i(a_i, b_i)$ denote the partial derivative $\frac{\partial F_i(a_i, b_i)}{\partial a_i}$.

Our generic transformation combines these ingredients into a randomized mechanism $\mathcal{M} = \texttt{AllocToMech}(\mathcal{A}, \mu, \mathbf{f})$ that works as follows:

1. It solicits bid vector $b \in T$,

[3]To keep the notation consistent, we state Definition 3.2 for a given agent i. Strictly speaking, the subscript i is not necessary.

Algorithm 1 The canonical self-resampling procedure.

```
1: function Recursive($b_i$)
2:     with probability $1-\mu$
3:         return $b_i$.
4:     else
5:         Pick $b'_i \in [0, b_i]$ uniformly at random.
6:         return Recursive($b'_i$).
7: Input: bid $b_i \in [0, \infty]$, parameter $\mu \in (0, 1)$.
8: Output: $(x_i, y_i)$ such that $0 \le x_i \le y_i \le b_i$.
9: with probability $1-\mu$
10:     $x_i \leftarrow b_i$, $y_i \leftarrow b_i$.
11: else
12:     Pick $b'_i \in [0, b_i]$ uniformly at random.
13:     $x_i \leftarrow$ Recursive($b'_i$), $y_i \leftarrow b'_i$.
```

2. It executes each agent's self-resampling procedure using an independent random seed w_i, to obtain two vectors of modified bids $x = (x_1(b_1; w_1), \ldots, x_n(b_n; w_n))$ and $y = (y_1(b_1; w_1), \ldots, y_n(b_n; w_n))$.

3. It then allocates[4] according to $\mathcal{A}(x)$.

4. Each agent i is charged the amount $b_i \cdot \mathcal{A}_i(x) - R_i$, where the *rebate* R_i is defined as

$$R_i = \begin{cases} \frac{1}{\mu} \cdot \frac{\mathcal{A}_i(x)}{F'_i(y_i, b_i)} & \text{if } y_i < b_i, \\ 0 & \text{otherwise}. \end{cases} \tag{5}$$

We are now ready to present our main result:

Theorem 3.3. *Let $\mathcal{A}$ be a monotone allocation rule. Suppose we are given an ensemble* $\mathbf{f}$ *of self-resampling procedures $f_i = (x_i, y_i)$ for each agent i, each with resampling probability $\mu \in (0, 1)$. Then the mechanism $\mathcal{M} = (\tilde{\mathcal{A}}, \tilde{\mathcal{P}}) = \texttt{AllocToMech}(\mathcal{A}, \mu, \mathbf{f})$ has the following properties.*

(a) $\mathcal{M}$ is truthful, universally ex-post individually rational,

(b) For n agents and any bid vector b (and any fixed random seed of nature) allocations $\tilde{\mathcal{A}}(b)$ and $\mathcal{A}(b)$ are identical with probability at least $1 - n\mu$.

(c) If $T = \Re^n_+$ (all types are positive), and each f_i is the canonical self-resampling procedure, then mechanism $\mathcal{M}$ is ex-post no-positive-transfers, and never pays any agent i more than $b_i \cdot \mathcal{A}_i(x) \cdot (\frac{1}{\mu} - 1)$.

Remark 3.4. *Several remarks are in order.*

(1) The mechanism never explicitly computes the payment for agent i (Eq. (4)) but rather implicitly creates the correct expected payments through its randomization of the bids.

(2) The mechanism only invokes the original allocation rule $\mathcal{A}$ once*. This property is very useful when it is impossible to invoke the allocation rule more than once, e.g. for multi-armed bandit allocations.*

[4]If $\mathcal{A}$ itself is randomized and/or if there is randomness arising from nature, then we allocate according to $\mathcal{A}(x; w, r)$ and we assume that w, r are independent of the random seeds w_i used in the resampling step.

(3) The mechanism $\mathcal{M}$ is randomized even if $\mathcal{A}$ is deterministic. It is truthful in expectation over the randomness used by the self-resampling procedures.

(4) If $\mathcal{A}$ is ex-post monotone, then $\mathcal{M}$ will be ex-post truthful. To see this, fix nature's random seed r and apply Theorem 3.3 to the allocation rule $\mathcal{A}_r$ induced by this r.

(5) If agents' types are positive then by part (b), the welfare of $\mathcal{M}$ is at least $1 - n\mu$ times that of $\mathcal{A}$. Parameter μ controls the trade-off between the loss in welfare and the size of the rebates R_i. Further results on bounding the welfare loss are presented in Section 3.5.

(6) By definition of the payment rule, the mechanism is universally ex-post normalized. We will not explicitly mention this property in the subsequent applications.

Proof of Theorem 3.3. To prove that $\mathcal{M}$ is truthful, we need to prove two things: that the randomized allocation rule $\tilde{\mathcal{A}}$ is monotone, and that the expected payment rule $\tilde{\mathcal{P}}$ satisfies

$$\tilde{\mathcal{P}}_i(b) = b_i \tilde{\mathcal{A}}_i(b_{-i}, b_i) - \textstyle\int_{-\infty}^{b_i} \tilde{\mathcal{A}}_i(b_{-i}, u)\, du. \tag{6}$$

Recall that when we write $\tilde{\mathcal{A}}_i(b_{-i}, u)$ without indicating the dependence on the combined random seed $q = (w_1, \dots, w_n, w, r)$ it means that we are referring to the unconditional expectation of $\tilde{\mathcal{A}}_i(b_{-i}, u; q)$.

The monotonicity of randomized allocation rule $\tilde{\mathcal{A}}$ follows from the monotonicity of $\mathcal{A}$ and the monotonicity property 1 in the definition of a self-resampling procedure. To prove that $\tilde{\mathcal{P}}_i$ satisfies Eq. (6), we begin by recalling that the payment charged to player i is $b_i \mathcal{A}_i(x) - R_i$, where the rebate R_i is defined by Eq. (5). The expectation of $b_i \mathcal{A}_i(x)$ is simply $b_i \tilde{\mathcal{A}}_i(b_{-i}, b_i)$, so to conclude the proof of truthfulness we must show that

$$\mathbb{E}[R_i] = \textstyle\int_{-\infty}^{b_i} \tilde{\mathcal{A}}_i(b_{-i}, u)\, du. \tag{7}$$

Our proof of Eq. (7) begins by observing that the conditional distribution of x_i, given that $y_i = u < b_i$, is the same as the unconditional distribution of $x_i(u; w_i)$, by Property 3 of a self-resampling procedure. Combining this with the fact that the random seed w_i is independent of $\{w_j : j \neq i\}$, we find that the conditional distribution of the tuple $x = (x_{-i}, x_i)$, given that $y_i = u$, is the same as the unconditional distribution of the vector $\hat{x}$ of modified bids that $\mathcal{M}$ would input into the allocation function $\mathcal{A}$ if the bid vector were (b_{-i}, u) instead of (b_{-i}, b_i). Taking expectations, this implies that for all $u < b_i$, we have $\mathbb{E}[\mathcal{A}_i(x) \,|\, y_i = u] = \mathbb{E}[\mathcal{A}_i(\hat{x})] = \tilde{\mathcal{A}}(b_{-i}, u)$.

Now apply Theorem 3.1 with the function $g(u) = \tilde{\mathcal{A}}_i(b_{-i}, u)$. Recalling that $F_i(\cdot, b_i)$ is the cumulative distribution function of y_i given that $y_i < b_i$, we apply the theorem to obtain

$$\begin{aligned}\int_{-\infty}^{b_i} \tilde{\mathcal{A}}_i(b_{-i}, u)\, du &= \mathbb{E}\left[\frac{\tilde{\mathcal{A}}_i(b_{-i}, y_i)}{F_i'(y_i, b_i)} \,\middle|\, y_i < b_i \right] = \mathbb{E}\left[\frac{\mathcal{A}_i(x)}{F_i'(y_i, b_i)} \,\middle|\, y_i < b_i \right] \\ &= \mu \cdot \mathbb{E}[R_i \,|\, y_i < b_i],\end{aligned} \tag{8}$$

where the second equation follows from the equation derived at the end of the preceding paragraph, averaging over all $u < b_i$. Observing that $R_i = 0$ unless $y_i < b_i$, an event that has probability μ, we see that $\mathbb{E}[R_i] = \mu \cdot \mathbb{E}[R_i \,|\, y_i < b_i]$. Combined with Eq. (8), this establishes Eq. (7) and completes the proof that $\mathcal{M}$ is truthful.

Mechanism $\mathcal{M}$ is universally ex-post individually rational because agent i is never charged an amount greater than $b_i \tilde{\mathcal{A}}_i(b; q)$. Part (b) follows from the union bound: the probability that $x_i = b_i$ for all i is at least $1 - n\mu$. For part (c), note that by Proposition 3.5, the canonical self-resampling procedure has distribution function $F(a_i, b_i) = a_i / b_i$, hence $F_i'(y_i, b_i) = 1/b_i$, for all i, y_i, b_i. The rebate R_i is equal either to 0 or to $\frac{1}{\mu} \cdot \frac{A_i(x)}{F_i'(y_i, b_i)} = b_i \cdot A_i(x) \cdot \frac{1}{\mu}$. We also charge $b_i \cdot A_i(x)$ to agent i. The claimed upper bound on the amount paid to agent i follows by combining these two terms. □

3.4 More on self-resampling procedures

First we analyze Algorithm 1, then we proceed to the case of general support.

Proposition 3.5. *Algorithm 1 is a self-resampling procedure with support $\Re_+$ and resampling probability μ. The distribution function for this procedure is $F(a_i, b_i) = a_i/b_i$.*

Proof Sketch. Properties 1 and 2 in Definition 3.2 are immediate from the description of the algorithm. Property 3 follows from the recursive nature of the sampling procedure: the event $y_i(b_i; w_i) = b'_i < b_i$ implies that the algorithm has followed the "else" branch on Line 11, and has chosen b'_i in Line 12. Finally, the distribution function is $F(a_i, b_i) = a_i/b_i$ since conditional on the event $y_i(b_i; w_i) < b_i$, the distribution of $y_i(b_i; w_i)$ is uniform in the interval $[0, b_i]$. Property 4 follows trivially. □

Recall that Algorithm 1 is recursive; henceforth let's call it RECURSIVE$_\mu$ for clarity. An equivalent explicit (non-recursive) version, called ONESHOT$_\mu$, can be stated as follows:

Algorithm 2 ONESHOT$_\mu$: a non-recursive version of RECURSIVE$_\mu$.

```
1: Input: bid $b_i \in [0, \infty]$, parameter $\mu \in (0, 1)$.
2: Output: $(x_i, y_i)$ such that $0 \le x_i \le y_i \le b_i$.
3: with probability $1 - \mu$
4:     $x_i \leftarrow b_i$, $y_i \leftarrow b_i$.
5: else
6:     Pick $\gamma_1, \gamma_2 \in [0, 1]$ indep., uniformly at random.
7:     $x_i \leftarrow b_i \cdot \gamma_1^{1/(1-\mu)}$,  $y_i \leftarrow b_i \cdot \max\{\gamma_1^{1/(1-\mu)}, \gamma_2^{1/\mu}\}$.
```

Proposition 3.6. RECURSIVE$_\mu$ *and* ONESHOT$_\mu$ *generate the same output distribution: for any bid $b_i \in [0, \infty)$, the joint distribution of the pair $(x_i, y_i) = (x_i(b_i; w_i), y_i(b_i; w_i))$ is the same for both procedures.*

This equivalence (whose proof is deferred to Appendix A) is further used for computing the integrals in the proofs of Theorem 3.3(c) and Theorem 4.1.

Now, to construct a self-resampling procedure with support in an arbitrary interval I, we can use the following technique. Suppose $h : (0, 1] \times I \to I$ is a two-variable function such that the partial derivatives $\partial h(z_i, b_i)/\partial z_i$ and $\partial h(z_i, b_i)/\partial b_i$ are well-defined and strictly positive at every point $(z_i, b_i) \in (0, 1] \times I$. Suppose furthermore that $h(1, b_i) = b_i$ and $\inf_{z_i \in (0,1]}\{h(z_i, b_i)\} = \inf(I)$ for all $b_i \in I$. Then we define the *h-canonical self-resampling procedure* (x_i^h, y_i^h) with support I, by specifying that

$$\begin{cases} x_i^h(b_i; w_i) &= h(x_i(1; w_i), b_i) \\ y_i^h(b_i; w_i) &= h(y_i(1; w_i), b_i), \end{cases} \tag{9}$$

where (x_i, y_i) is the canonical self-resampling procedure as defined in Algorithm 1.

Proposition 3.7. *(x_i^h, y_i^h) as defined in Eq. (9) is a self-resampling procedure with support I and resampling probability μ. The distribution function for (x_i^h, y_i^h) is the unique two-variable function $F(a_i, b_i)$ such that*

$$h(F(a_i, b_i), b_i) = a_i \quad \textit{for all } a_i, b_i \in I,\ a_i < b_i. \tag{10}$$

Proof. Property 1 in Definition 3.2 holds because of the monotonicity of h, Property 2 holds because $h(1, b_i) = b_i$ for all b_i, and Property 3 holds because the function h is deterministic and monotone.

Let $F_h(a_i, b_i)$ and $F_0(a_i, b_i)$ be the distribution functions for the h-canonical and canonical self-resampling procedures, respectively. Recall that $F_0(a_i, b_i) = a_i/b_i$ by Proposition 3.5. Note that $F(a_i, b_i)$ in Eq. (10) is unique (and hence well-defined) by the strict monotonicity of h.

The claim that $F_h(a_i, b_i) = F(a_i, b_i)$ easily follows from in Eq. (9). By definition of h it holds that

$$h(y_i(1, w_i),\, b_i) < b_i \iff y_i(1, w_i) < 1.$$

Therefore, letting $y_i = y_i(1, w_i)$ we have

$$\begin{aligned} F_h(a_i, b_i) &\triangleq \Pr[h(y_i, b_i) < a_i \,|\, h(y_i, b_i) < b_i] \\ &= \Pr[y_i < F(a_i, b_i) \,|\, y_i < 1] \\ &= F_0(\, F(a_i, b_i)\, ,\, 1) \\ &= F(a_i, b_i). \end{aligned}$$

Our assumption that h is differentiable and strictly increasing in its first argument now implies that the same property holds for F, which verifies Property 4. □

3.5 Bounds on welfare

In this section we present bounds on the welfare obtained by our generic transformation for two interesting special cases: the positive-only types and the negative-only types. We consider the approximation that is achieved by the mechanism as a function of the approximation of the original allocation rule. As per Theorem 3.3(b), the generic transformation creates a mechanism with an allocation that is identical to the original allocation with probability at least $1 - n\mu$. For positive-only types this implies a bound on the approximation which degrades with n, the number of agents (see Remark 3.4(5)). For negative-only types such bound does not immediately imply since the cost in the low probability event might be prohibitively high. For both setting, we present similar bounds that do *not* degrade with n.

Positive-only types. We first consider the generic procedure for positive-only types: $T = \Re_+^n$, where $\Re_+ = (0, \infty)$. Recall that for agents' types $t \in T$ the social welfare of an outcome o is defined to $SW(o, t) = \sum_{i \in N} t_i\, a_i(o)$. The optimal social welfare is $OPT(t) = \max_{o \in O} SW(o, t)$, where O is the set of all feasible outcomes. (A mechanism with) an allocation rule $\mathcal{A}$ is α-*approximate* if it holds that

$$\alpha \cdot \mathbb{E}[SW(\mathcal{A}(t), t)] \geq OPT(t) \text{ for every } t. \tag{11}$$

Theorem 3.8. *Consider the setting in Theorem 3.3(c), so that $T = \Re_+^n$ and each f_i is the canonical self-resampling procedure. If allocation rule $\mathcal{A}$ is α-approximate, then mechanism* `AllocToMech`$(\mathcal{A}, \mu, \mathbf{f})$ *is $\alpha/(1 - \frac{\mu}{2-\mu})$-approximate.*

Proof. Fix a bid vector b, and let o^* be the corresponding optimal allocation. Recall that our mechanism outputs allocation $\mathcal{A}(x)$, where x is the vector of randomly modified bids. As the original allocation rule $\mathcal{A}$ is α-approximate, by Eq. (11) it holds that $\alpha \cdot SW(\mathcal{A}(x), x) \geq OPT(x)$. We show (see Lemma 3.10 below) that

$$\mathbb{E}[x_i] = \left(1 - \tfrac{\mu}{2-\mu}\right) b_i \text{ for each agent } i. \tag{12}$$

Thus when we evaluate o^* with respect to bids x we get:

$$\begin{aligned}\alpha \cdot \mathbb{E}[SW(A(x),x)] &\geq OPT(x)\\ &\geq SW(o^*,x)\\ &= \textstyle\sum_{i\in N} x_i\, a_i(o^*)\\ &= \sum_{i\in N} \left(1-\tfrac{\mu}{2-\mu}\right) b_i\, a_i(o^*)\\ &= \left(1-\tfrac{\mu}{2-\mu}\right) OPT(b). \qquad \square\end{aligned}$$

Remark 3.9. *For arbitrary self-resampling procedures f_i with support $\Re_+$, Eq. (12) can be replaced by $\mathbb{E}[x_i] \geq (1-\mu)\, b_i$, which gives a slightly weaker result, namely an $\alpha/(1-\mu)$-approximation to the social welfare.*

Lemma 3.10. *In the setting of Theorem 3.8, letting x be the vector of modified types, Eq. (12) holds.*

Proof. Let us use $\textsc{OneShot}_\mu$ to describe the canonical self-resampling procedure. Recall that $\textsc{OneShot}_\mu$ generates $x_i = x_i(b_i; w_i)$ by setting $x_i = b_i$ with probability $1-\mu$, and otherwise sampling γ_1 uniformly at random in $[0,1]$ and outputting $x_i = b_i \cdot \gamma_1^{1/(1-\mu)}$. Hence

$$\begin{aligned}\mathbb{E}[x_i \,|\, x_i < b_i] &= \textstyle\int_0^1 b_i \cdot \gamma_1^{1/(1-\mu)}\, d\gamma_1 = b_i \cdot \frac{1}{1+\frac{1}{1-\mu}} = b_i \cdot \left(1 - \frac{1}{2-\mu}\right)\\ \mathbb{E}[x_i] &= (1-\mu)\cdot b_i + \mu \cdot \mathbb{E}[x_i \,|\, x_i < b_i] = b_i \cdot \left(1-\tfrac{\mu}{2-\mu}\right). \qquad \square\end{aligned}$$

Negative-only types. We next consider the negative-only types setting: $T = \Re_-^n$, where $\Re_- = (-\infty, 0)$.

For negative types approximation is defined with respect to the social cost, which is the negation of the social welfare. An algorithm is α*-approximate* if for every input it outputs an outcome with cost at most α times the optimal cost. We present an approximation bound for an h-canonical self-resampling procedure, for a suitably chosen h.

Theorem 3.11. *Consider the setting in Theorem 3.3. Assume that $T = \Re_-^n$ and that each f_i is the h-canonical self-resampling procedure, where $h(z_i, b_i) = b_i/\sqrt{z_i}$. Suppose $\mu \in (0, \frac{1}{2})$. If allocation rule $\mathcal{A}$ is α-approximate, then mechanism $\texttt{AllocToMech}(\mathcal{A}, \mu, \mathbf{f})$ is $\alpha\left(1+\frac{\mu}{1-2\mu}\right)$-approximate.*

The proof of this theorem is almost identical to that of Theorem 3.8, and thus is omitted. The main modification is that Lemma 3.10 is replaced by the following lemma:

Lemma 3.12. *In the setting of Theorem 3.11, letting x^h be the vector of modified types, it holds that*

$$\mathbb{E}[x_i^h] = b_i \left(1+\tfrac{\mu}{1-2\mu}\right) \text{ for all } i.$$

Proof. Recall that x^h is defined by Eq. (9). As in Lemma 3.10, we will use $\textsc{OneShot}_\mu$ to describe the canonical self-resampling procedure. It follows that

$$\begin{aligned}\mathbb{E}[x_i^h \,|\, x_i < b_i] &= \int_0^1 \frac{b_i}{\sqrt{\gamma_1^{\frac{1}{(1-\mu)}}}}\, d\gamma_1 = \int_0^1 b_i \cdot \gamma_1^{-\frac{1}{2(1-\mu)}}\, d\gamma_1 = b_i \cdot \frac{1}{1-\frac{1}{2(1-\mu)}} = b_i \cdot \left(1+\frac{1}{1-2\mu}\right),\\ \mathbb{E}[x_i^h] &= (1-\mu)\cdot b_i + \mu \cdot \mathbb{E}[x_i^h \,|\, x_i^h < b_i] = b_i \cdot \left(1+\tfrac{\mu}{1-2\mu}\right). \qquad \square\end{aligned}$$

4 Some Applications

The VCG mechanism for shortest paths. The seminal paper of Nisan and Ronen [15] has presented the following question: is there a computational overhead in computing payments that will induce agents to be truthful, compared to the computation burden of computing the allocation. One of their examples is the VCG mechanism for the *shortest path mechanism design problem*, where a naive computation of VCG payments requires additional computation of n shortest path instances. Yet, an explicit payment computation is not the real goal, it is just a means to an end. The real goal is *inducing the right incentives*. Our procedure shows that without *any* overhead in computation, if we move to a randomized allocation rule and settle for truthfulness in expectation (and an approximately efficient allocation) one can induce the right incentives.

Specifically, the shortest path mechanism design problem is the following. We are given a graph $G = (V, E)$ and a pair of source-target nodes (v_s, v_t). Each agent e controls an edge $e \in E$ and has a cost $c_e > 0$ if picked (thus $v_e = -c_e < 0$ and $T_e = (-\infty, 0)$ for every e). That cost is private information, known only to agent e. The mechanism designer's goal is to pick a path P from node v_s to node v_t in the graph with minimal total cost, that is $\sum_{e \in P} c_e$ is minimal. Assume that there is no edge that forms a cut between v_s and v_t.

The VCG mechanism is an efficient and truthful mechanism for this problem. It computes a shortest path P with respect to the reported costs and pays to an agent e the difference between the cost of the shortest path that does not contains e and the total cost shortest path excluding the cost of e. A naive implementation of the VCG mechanism requires computing $|P| + 1$ shortest path instances (where $|P|$ denotes the number of edges in path P). VCG is deterministic, truthful and efficient (1-approximation).

Let EFF an the efficient allocation rule for the shortest path problem. We can use our general procedure to derive the following result (it's proof follows directly from Theorem 3.3 and the application of Theorem 3.11 for EFF which is an the efficient allocation rule and thus is 1-approximation).

Theorem 4.1. *Fix any $\mu \in (0, \frac{1}{2})$. For each agent i, let f_i be the h-canonical self-resampling procedure, where $h(z_i, b_i) = b_i/\sqrt{z_i}$. Let $\mathcal{M} = \texttt{AllocToMech}(\texttt{EFF}, \mu, \{f_i\})$ be the mechanism created by applying $\texttt{AllocToMech}()$ to EFF. Then $\mathcal{M}$ has the following properties:*

- *It is truthful and universally individually rational.*
- *It only computes one shortest paths instance.*
- *It outputs a path with expected length at most $\left(1 + \frac{\mu}{1-2\mu}\right)$ times the length of the shortest path.*

Remark 4.2. *Recall that parameter μ controls the trade-off between approximation ratio and the rebate size R_i, which for a given random seed is proportional to $\frac{1}{\mu}$.*

Communication overhead of payment computation. The paper [7] shows that there exists a monotone deterministic allocation rule for which the communication required for computing the allocation is factor $\Omega(n)$ less than the communication required to computing prices. This implies that inducing the correct incentives *deterministically* has a large overhead in communication. Assume that instead of requiring explicit computation of payments we are satisfied with inducing the correct incentives using a *randomized* mechanism. In such case our reduction shows that the deterministic lower bound cannot be extended to randomized mechanisms, if we allow a small error in the allocation.

More concretely, consider a single parameter domain with types that are positive, $T_i = (0, \infty)$ (as in [7]). For all i, use the canonical self-resampling procedure. Consider any monotone allocation rule $\mathcal{A}$. We can apply Theorem 3.3 to obtain a randomized mechanism that is truthful and only executes that allocation rule $\mathcal{A}$ *once* (thus has no communication overhead at all) and has exactly the same allocation with probability at least $(1 - \mu)^n$. For any $\epsilon > 0$ we can find $\mu > 0$ such that the error probability is less than ϵ.

5 Multi-armed bandit mechanisms

In this section we apply the main result to *multi-armed bandit (MAB) mechanisms*: single-parameter mechanisms in which the allocation rule is (essentially) an MAB algorithm parameterized by the bids. As in any single-parameter mechanism, agents submit their bids, then the allocation rule is run, and then the payments are assigned. This application showcases the full power of the main result, since in the MAB setting the allocation rule is only run once, and (in general) cannot be simulated as a computational routine without actually implementing the allocation.

Focusing on the stochastic setting, we design truthful MAB mechanisms with the same regret guarantees as the best MAB *algorithms* such as `UCB1` [5]. First, we prove that allocation rules derived from `UCB1` and similar MAB algorithms are in fact monotone, and hence give rise to truthful MAB mechanisms. Second, we provide a new allocation rule with the same regret guarantees that is ex-post monotone, and hence gives rise to an *ex-post* truthful MAB mechanism. Third, we use this new allocation rule to obtain an unconditional separation between the power of randomized and deterministic ex-post truthful MAB mechanisms.

5.1 Preliminaries: MAB mechanisms

An MAB mechanism [8, 10] operates as follows. There are n agents. Each agent i has a private value v_i and submits a bid b_i. We assume that $b_i, v_i \in [0, b_{\text{max}}]$, where b_{max} is known a priori. The allocation consists of T rounds, where T is the *time horizon*. In each round t the allocation rule chooses one of the agents, call it $i = i(t)$, and observes a *click reward* $\pi(t) \in [0,1]$; the chosen agent i receives $v_i\,\pi(t)$ units of utility. Payments are assigned after the last round of the allocation. Note that the social welfare of the mechanism is equal to the total value-adjusted click reward: $\sum_{t=1}^{T} v_{i(t)}\,\pi(t)$.

The special case of 0-1 click rewards corresponds to the scenario in which agents are advertisers in a pay-per-click auction, and choosing agent i in a given round t means showing this agent's ad. Then the click reward $\pi(t)$ is the *click bit*: 1 if the ad has been clicked, and 0 otherwise. Following the web advertising terminology, we will say that in each round, an *impression* is allocated to one of the agents.

Formally, an *MAB allocation rule* $\mathcal{A}$ is an online algorithm parameterized by n, T, b_{max} and the bids b. In each round it allocates the impression and observes the click reward. Absent truthfulness constraints, the objective is to maximize the *reported welfare*: $\sum_{t=1}^{T} b_{i(t)}\,\pi(t)$. This formulation generalizes MAB *algorithms*: the latter are precisely MAB allocation rules with all bids set to 1.

Given an MAB algorithm $\hat{\mathcal{A}}$, there is a natural way to transform it into an MAB allocation rule $\mathcal{A}$. Namely, $\mathcal{A}$ runs algorithm $\hat{\mathcal{A}}$ with modified click rewards: if agent i is chosen in round t then the click reward reported to $\hat{\mathcal{A}}$ is $\hat{\pi}(t) = (b_i/b_{\text{max}})\,\pi(t)$. We will say that algorithm $\hat{\mathcal{A}}$ *induces* allocation rule $\mathcal{A}$. From now on we will identify an MAB algorithm with the induced allocation rule, e.g. allocation rule `UCB1` is induced by algorithm `UCB1` [5].

We will focus on the *stochastic* MAB setting: in all rounds t in which an agent i is chosen, the click reward $\pi(t)$ is an independent random sample from some fixed distribution on $[0,1]$ with expectation μ_i.[5] Following the web advertisement terminology, we will call μ_i the *click-through rate* (CTR) of agent i. The CTRs are fixed, but no further information about them (such as priors) is revealed to the mechanism.

Regret. The performance of an MAB allocation rule is quantified in terms of *regret*:

$$R(T;b;\mu) \triangleq T\max_i[\,b_i\,\mu_i\,] - \mathbb{E}[\,\textstyle\sum_{t=1}^{T} b_{i(t)}\,\mu_{i(t)}\,],$$

the difference in expected click rewards between the algorithm and the benchmark: the best agent in hindsight, knowing the μ_i's. We focus on $R(T) \triangleq \max R(T;b;\mu)$, where the maximum is taken over all CTR

[5]The exact shape of this distribution is not essential. E.g. in the advertising example $\pi(t) \in \{0,1\}$.

vectors μ and all bid vectors b such that $b_i \leq 1$ for all i. [6]

Regret guarantees from the vast literature on MAB algorithms easily translate to MAB allocation rules. In particular, allocation rule `UCB1` has regret $R(T) = O(\sqrt{nT \log T})$ [5], which is nearly matching the information-theoretically optimal regret bound $\Theta(\sqrt{nT})$ [6, 4]. The stochastic MAB setting tends to be easier if the best agent is much better than the second-best one. Let us sort the agents so that $b_1\,\mu_1 \geq b_2\,\mu_2 \geq \ldots \geq b_n\,\mu_n$. The *gap* δ of the problem instance is defined as $(b_1\,\mu_1 - b_2\,\mu_2)/b_{\max}$. The *$\delta$-gap regret* $R_\delta(T)$ is defined as the worst-case regret over all problem instances with gap δ. Allocation rule `UCB1` achieves $R_\delta(T) = O(\frac{n}{\delta}\,\log T)$ [5]; there is a lower bound $R_\delta(T) = \Omega(\min(\frac{n}{\delta}\,\log T,\ \sqrt{nT}))$ [13, 6, 12].

Click realizations. A *click realization* is a $k \times T$ table ρ in which the (i, t) entry $\rho_i(t)$ is the click reward (e.g., the click bit) that agent i receives if it is played in round t. Note that in order to fully define the behavior of any algorithm on all bid vectors one may need to specify all entries in the table, whereas only a subset thereof is revealed in any given run. We view ρ as a realization of nature's random seed. Thus, we can now define ex-post truthfulness and other ex-post properties: informally, ex-post property is a property that holds for every given click realization.

For each agent i, round t, bid vector b and click realization ρ, let $\mathcal{A}_i^t(b; \rho)$ (resp., $\hat{\mathcal{A}}_i^t(\rho)$) denote the probability that MAB allocation rule $\mathcal{A}$ (resp., MAB algorithm $\hat{\mathcal{A}}$) allocates the impression at round t to agent i.

5.2 Truthfulness and monotonicity

Theorem 3.3(c) reduces the problem of designing truthful MAB mechanisms to that of designing monotone MAB allocations. Let us state this reduction explicitly:

Theorem 5.1. *Consider the stochastic MAB mechanism design problem. Let $\mathcal{A}$ be a stochastically monotone (resp., ex-post monotone) MAB allocation rule. Applying the transformation in Theorem 3.3(c)[7] to $\mathcal{A}$ with parameter μ, we obtain a mechanism $\mathcal{M}$ such that:*

*(a) $\mathcal{M}$ is **stochastically truthful** (resp., **ex-post truthful**), ex-post no-positive-transfers, and universally ex-post individually rational.*

(b) for each click realization, the difference in expected welfare between $\mathcal{A}$ and $\mathcal{M}$ is at most $\mu n\, b_{max}$.

Remark 5.2. *The theorem provides two distinct types of guarantees: game-theoretic guarantees in part (a), and performance guarantees in part (b).*

Remark 5.3. *The performance guarantee in part (b) depends on n, the number of agents. If instead of expected welfare one considers regret $R(T)$, the dependence on n can be eliminated, so that the difference between $\mathcal{A}$ and $\mathcal{M}$ is at most μ. In fact, this result extends to regret (stochastic or ex-post) w.r.t. any given benchmark set of outcomes O'. To prove this, use (an additive version of) Theorem 3.8, with OPT redefined w.r.t. O'; we omit the details.*

We show that a very general class of deterministic MAB algorithms induces monotone MAB allocation rules (to which Theorem 5.1 can be applied).

Definition 5.4. *In a given run of an MAB algorithm, the* round-t statistics *is a pair of vectors (π, ν), where the i-th component of π (resp., ν) is equal to the total payoff (resp., the number of impressions) of agent i in rounds 1 to $t-1$, for each agent i. Vectors π and ν are called* p-stats vector *and* i-stats vector*, respectively.*

[6]We define $R(T)$ with $b_{\max} = 1$ merely to simplify the notation. All regret bounds (scaled up by a factor of $b_{\max}$) hold for an arbitrary $b_{\max}$.

[7]Theorem 3.3(c) is stated for $T = \Re_+^n$, but it trivially extends to the case $T = (0, b_{\max})^n$.

Definition 5.5. *A deterministic MAB algorithm $\hat{\mathcal{A}}$ is called* **well-formed** *if for each round t and each agent i, letting (π, ν) be the round-t statistics, the following properties hold:*

- *$\hat{\mathcal{A}}_i^t(\rho)$ is determined by (π, ν): $\hat{\mathcal{A}}_i^t(\rho) = \chi_i(\pi; \nu)$ for any click realization ρ.* [8]
- *[χ-monotonicity] $\chi_i(\pi; \nu)$ is non-decreasing in π_i for any fixed (π_{-i}, ν).*
- *[χ-IIA] for each round t, any three distinct agents $\{i, j, l\}$ and any fixed (π_{-i}, ν_{-i}), changing (π_i, ν_i) cannot transfer an impression from j to l.*

The χ-*IIA* property above is reminiscent of *Independence of Irrelevant Alternatives* (*IIA*) property in the Social Choice literature (hence the name). A similar but technically different property is essential in the analysis of deterministic MAB allocation rules in [8].

Remark 5.6. *For a concrete example of a well-formed MAB algorithm, consider (a version of)* `UCB1`.[9] *The algorithm is very simple: in each round t, it chooses agent*

$$\min\left(\arg\max_i \left(\pi_i(t)/\nu_i(t) + \sqrt{8\log(T)/\nu_i(t)}\right)\right).$$

Lemma 5.7. *In the stochastic MAB mechanism design problem, let $\mathcal{A}$ be a MAB allocation rule induced by a well-formed MAB algorithm. Then $\mathcal{A}$ is stochastically monotone.*

Proof. We will use an alternative way to define a realization of random click rewards: a *stack-realization* is a $k \times T$ table in which the (i, t) entry is the click bit that agent i receives the t-th time she is played. Clearly a stack-realization and a bid vector uniquely determine the behavior of $\mathcal{A}$. We will show that:

$$\mathcal{A} \text{ is monotone for each stack-realization.} \tag{13}$$

Then $\mathcal{A}$ is monotone in expectation over any distribution over stack-realizations, and in particular it is monotone in expectation over the random clicks in the stochastic MAB setting, so the Lemma follows.

Let us prove Claim (13). Throughout the proof, fix stack-realization σ, agent i, and bid vector b_{-i}. Consider two bids $b_i < b_i^+$. The claim asserts that agent i receives at least as many clicks with bid b_i^+ than with bid b_i.

Let us introduce some notation. Given stack-realization σ and bid vector $b = (b_{-i}, b_i)$, let $\mathcal{A}(b_i, t)$ be the agent selected by the allocation rule in round t, and let $\nu_i(b_i, t)$ (resp., $\pi_i(b_i, t)$) be the total number of impressions (resp., total click reward) of agent i in the first t rounds. Let $\hat{\pi}_i(b_i, t) = (b_i/b_{\text{max}})\ \pi_i(b_i, t)$ be the corresponding total *modified* click rewards. Note that $\nu_i(b_i, t)$ uniquely determines $\pi_i(b, t)$:

$$\pi_i(b_i, t) = \textstyle\sum_{s=1}^{\nu_i} \sigma(i, s) \quad \text{where} \quad \nu_i = \nu_i(b_i, t). \tag{14}$$

Let us overview the forthcoming technical argument. We will show by induction on t that $\nu_i(b_i, t) \leq \nu_i(b_i^+, t)$ for all t. For the induction step we only need to worry about the case when the claim holds for a given t with *equality*. In this case we show that $\nu_{-i}(b_i, t) = \nu_{-i}(b_i^+, t)$. This is trivial for $n = 2$ agents; the general case requires a rather delicate argument that uses the χ-IIA property in Definition 5.5.[10]

[8] As the algorithm is deterministic, the probability $\hat{\mathcal{A}}_i^t(\rho)$ is trivial: either 0 or 1. The function $\chi_i()$ is the same for all t.

[9] To ensure the χ-IIA property, we use a slightly modified version of `UCB1`: $\log T$ is used instead of $\log t$, and $\min$ is used to break ties (instead of an arbitrary rule). This change does not affect regret guarantees. We will denote this version as `UCB1` without further notice.

[10] Also, we will use the fact that the probabilities $\chi_j(\hat{\pi}, \nu)$ in Definition 5.5 do not depend on the round (given j and $(\hat{\pi}, \nu)$). This is the only place in any of the proof where we invoke this fact.

Now let us carry out the proofs in detail. First, denote $\nu_*(b_i,t) \triangleq t - \nu_i(b_i,t)$, and let us show that for any two rounds t, s it holds that

$$\nu_*(b_i,t) = \nu_*(b_i^+,s) \;\Rightarrow\; \nu_{-i}(b_i,t) = \nu_{-i}(b_i^+,s). \tag{15}$$

Let us use induction on $\nu_*(b_i,t)$. For $\nu_*(b_i,t) = 0$ the statement is trivial. For the induction step, suppose Eq. (15) holds whenever $\nu_*(b_i,t) = \nu_*$, and let us suppose $\nu_*(b_i,t) = \nu_*(b_i^+,s) = \nu_* + 1$. Let t' and s' be the latest rounds such that $\nu_*(b_i,t') = \nu_*(b_i^+,s') = \nu_*$. By the induction hypothesis, $\nu_{-i}(b_i,t') = \nu_{-i}(b_i^+,s')$. It remains to prove that $\mathcal{A}(b_i,t'+1) = \mathcal{A}(b_i^+,s'+1)$, i.e. that the allocation rule's selections in round $t'+1$ given bids (b_{-i},b_i), and in round $s'+1$ given bids (b_{-i},b_i^+), are the same.[11] By Definition 5.5 these selections are uniquely determined (given the stack-realization) by the bids and the impression counts ν. By the choice of t' and s', neither of the two selections is i, so by the χ-IIA condition in Definition 5.5 the selections are uniquely determined by b_{-i} and ν_{-i}, and hence are the same. This proves Eq. (15).

Now, to prove Claim (13) it suffices to show that for all t

$$\nu_i(b_i,t) \le \nu_i(b_i^+,t). \tag{16}$$

Let us use induction on t. The claim is trivial for $t = 1$, since the click bit of agent i in round 1 does not depend on $(b;\sigma)$. For the induction step, assume that the assertion Eq. (16) holds for some t, and let us prove it for $t+1$. Note that (using the notation from Definition 5.5)

$$\nu_i(b_i,t+1) = \nu_i(b_i,t) + \chi_i(\hat{\pi}(b_i,t);\ \nu(b_i,t)).$$

Now, $\nu_i(b_i,t) \le \nu_i(b_i^+,t)$ by induction hypothesis. If the inequality is strict then Eq. (16) trivially holds for $t+1$. Now suppose $\nu_i(b_i,t) = \nu_i(b_i^+,t)$. Then by Eq. (15) we have $\nu(b_i,t) = \nu(b_i^+,t)$. Moreover, by Eq. (14) we have $\pi(b_i,t) = \pi(b_i^+,t)$ and therefore $\hat{\pi}_{-i}(b_i,t) = \hat{\pi}_{-i}(b_i^+,t)$ and $\hat{\pi}_i(b_i,t) < \hat{\pi}_i(b_i^+,t)$. Thus, by the χ-monotonicity property in Definition 5.5 we have

$$\chi_i(\hat{\pi}(b_i,t);\ \nu(b_i,t)) \le \chi_i(\hat{\pi}(b_i^+,t);\ \nu(b_i^+,t)).$$

This concludes the proof of Eq. (16), and that Claim (13). ☐

5.3 Truthfulness vs regret

In this subsection we focus on the stochastic MAB setting, and consider the trade-off between regret and various notions of truthfulness. Ideally, one would like an MAB mechanism to be truthful in the strongest possible sense (universally ex-post), and have the same regret bounds as optimal MAB *algorithms*.

Let us start with some background. In [8] it was proved that any *deterministic* mechanism that is ex-post truthful and ex-post normalized (under very mild restrictions), and any distribution over such deterministic mechanisms, incurs much higher regret than an optimal MAB algorithm such as `UCB1`. Namely, the lower bound in [8] states that $R(T) = \Omega(n^{1/3}\,T^{2/3})$, whereas `UCB1` has regret $R(T) = O(\sqrt{nT\log T})$. [12] For δ-gap instances the difference is even more pronounced: the analysis in [8] provides a polynomial lower bound of $R_\delta(T) = \Omega(\delta\,T^\lambda)$ for some $\lambda > 0$, whereas `UCB1` achieves *logarithmic* regret $R_\delta(T) = O(\frac{n}{\delta}\log T)$.

Our first result is that we can use the machinery from Section 5.2 to match the regret of `UCB1` for truthful mechanisms. We apply Theorem 5.1 (with $\mu = \frac{1}{T}$) and Lemma 5.7 to `UCB1` to obtain the following corollary:

[11]Then $\nu_{-i}(b_i,t) = \nu_{-i}(b_i^+,s)$ because in all rounds from $t'+2$ to t (resp., from $s'+2$ to s) agent i is played.

[12]Following the literature on regret minimization, we are mainly interested in the asymptotic behavior of $R(T)$ as a function of T when n is fixed.

Corollary 5.8. *In the stochastic MAB mechanism design problem, there exists a mechanism $\mathcal{M}$ such that*

*(a) $\mathcal{M}$ is **stochastically truthful**, ex-post no-positive-transfers, universally ex-post individually rational.*

(b) $\mathcal{M}$ has regret $R(T) = O(\sqrt{nT \log T})$ and δ-gap regret $R_\delta(T) = O(\frac{n}{\delta} \log T)$.

Remark 5.9. *The regret and δ-gap regret in the above theorem are within small factors (resp., $O(\sqrt{\log T})$ and $O(1)$) of the best possible for any MAB allocation rule.*

Remark 5.10. *[8] provides a weaker result which transforms any monotone MAB algorithm such as* `UCB1` *into a truthful and normalized MAB mechanism with matching regret bounds. The guarantees in [8] are weaker for the following reasons. First, it only applies to 0-1 click rewards, whereas our setting allows for arbitrary click rewards in $[0, 1]$. Second, the individual rationality guarantee in [8] is much weaker: an agent may be charged more than her bid (which never happens in our mechanism), and the charge may be huge, as high as $b_i \times (4n)^T$; thus, a risk-averse agent may be reluctant to participate. Third, the no-positive-transfers guarantee is weaker: for some realizations of the click rewards the expected payment may be negative. Finally, the payment rule in [8] requires (as stated) a prohibitively expensive computation.*

The truthfulness in Corollary 5.8 is only in expectation over the random click rewards. Thus, after seeing a specific realization of the rewards an agent might regret having been truthful. Accordingly, we would like a stronger property: ex-post truthfulness, i.e. truthfulness *for every given realization of the rewards*.

The main result of this section is an ex-post truthful MAB mechanism with optimal regret bounds. Unlike Corollary 5.8, this result requires designing a new MAB allocation rule. This allocation rule and its analysis are the main technical contributions.

Theorem 5.11. *In the stochastic MAB mechanism design problem, there is a mechanism $\mathcal{M}$ such that*

*(a) $\mathcal{M}$ is **ex-post truthful**, ex-post no-positive-transfers, and universally ex-post individually rational.*

(b) $\mathcal{M}$ has regret $R(T) = O(\sqrt{nT \log T})$ and δ-gap regret $R_\delta(T) = O(\frac{n}{\delta} \log T)$.

The theorem follows from Theorem 5.1 (with $\mu = \frac{1}{T}$) if there exists an MAB allocation rule that is ex-post monotone and has the claimed regret bounds. To the best of our knowledge, such allocation rule is absent in the literature. Below we provide such allocation rule, called `NewCB`.

`NewCB` maintains a set of active agents; initially all agents are active. For each round t, there is a *designated agent* $i = 1 + (t \bmod n)$. If this agent is active, then it is allocated. Else, an active agent is chosen at random and allocated. For each agent i, lower and upper confidence bounds (L_i, U_i) on the product $b_i\, \mu_i$ are maintained (recall that μ_i is the CTR of agent i). After each round, each agent is de-activated if its upper confidence bound is smaller than someone else's lower confidence bound. The pseudocode is in Algorithm 3.

Fix realization ρ and bid vector b. Let $S_{\texttt{act}}(t, b)$ be the set of active agents after round t. For each agent i, let $L_i(t, b)$ and $U_i(t, b)$ be the values of L_i and U_i after round t.

The goal of the specific update rules for the confidence bounds (lines 15-20) and the statistics (lines 13) is to guarantee the following two properties:

- the statistics are kept only for rounds when a designated agent is played. Moreover, for each agent i and round t, and any two bid vectors b and b' we have

$$\text{if } i \in S_{\texttt{act}}(t, b) \cap S_{\texttt{act}}(t, b') \text{ then } \left\{ \begin{array}{lcl} L_i(t, b)/b_i & = & L_i(t, b')/b'_i \\ U_i(t, b)/b_i & = & U_i(t, b')/b'_i \end{array} \right. . \qquad (17)$$

Algorithm 3 `NewCB`: ex-post monotone MAB allocation rule.

```
1: Given: $n$ = #agents, $T$ = #rounds, upper bound $b_{\max}$.
2: Solicit a bid vector $b$ from the agents; $b \leftarrow b/b_{\max}$.
3: Initialize: set of active agents $S_{\texttt{act}} = \{$all agents$\}$.
4: for all agent $i$ do
5:    $c_i \leftarrow 0$; $n_i \leftarrow 0$ {total click reward and #impressions}
6:        {the totals are only over "designated" rounds}
7:    $U_i \leftarrow b_i$; $L_i \leftarrow 0$ {Upper and Lower Confidence Bounds}
8: {Main Loop}
9: for rounds $t = 1, 2, \ldots, T$ do
10:    $i \leftarrow 1 + (t \bmod n)$. {The "designated" agent}
11:    if $i \in S_{\texttt{act}}$ then
12:       Allocate agent $i$.
13:       $n_i \leftarrow n_i + 1$; $c_i \leftarrow c_i + \texttt{reward}$. {Update statistics.}
14:       {Update confidence bounds.}
15:       if $L_i < U_i$ then
16:          $(L'_i,\, U'_i) \leftarrow b_i\,(c_i/n_i \mp \sqrt{8\log(T)/n_i})$.
17:          if $\max(L_i,\, L'_i) < \min(U_i,\, U'_i)$ then
18:             $(L_i,\, U_i) \leftarrow (\max(L_i,\, L'_i),\ \min(U_i,\, U'_i))$.
19:          else
20:             $(L_i,\, U_i) \leftarrow (\frac{L_i+U_i}{2},\ \frac{L_i+U_i}{2})$.
21:    else
22:       Allocate an agent chosen u.a.r. from $S_{\texttt{act}}$.
23:    for all agent $i \in S_{\texttt{act}}$ do
24:       if $U_i < \max_{j \in S_{\texttt{act}}} L_j$ then
25:          Remove $i$ from $S_{\texttt{act}}$.
```

- for any fixed realization ρ and bid vector b, and each agent i: $L_i \leq U_i$, and from round to round L_i is non-decreasing and U_i is non-increasing. On other words, for each round t it holds that

$$L_i(t-1, b) \leq L_i(t, b) \leq U_i(t, b) \leq U_i(t-1, b). \tag{18}$$

The ex-post monotonicity follows from these two properties and the de-activation rule (lines 24-25).

Ex-post monotonicity. Let $L^*(t,b) \triangleq \max_{i \in S_{\texttt{act}}(t,b)} L_i(t,b)$. Fix agent i and $b_i^+ > b_i$, and let $b^+ = (b_{-i},\, b_i^+)$ be the "alternative" bid vector.

Claim 5.12. *We establish the following sequence of claims:*
(C1) $L^*(t,b)$ is non-decreasing in t, for any fixed b.
(C2) For each round t, $L^*(t,b) \leq L^*(t,b^+)$.
(C3) For each round t, $S_{\texttt{act}}(t,b^+) \setminus \{i\} \subset S_{\texttt{act}}(t,b) \setminus \{i\}$.
(C4) In each round t: if $i \in S_{\texttt{act}}(t,b)$ then $i \in S_{\texttt{act}}(t,b^+)$.

Proof. Let us prove the parts (C1-C4) one by one.

(C1). We use Eq. (18) and the de-activation rule. Throughout the proof, we omit the b. Fix round $t \geq 2$. Let $i \in S_{\texttt{act}}(t-1)$ be an agent such that $L^*(t-1) = L_i(t-1)$. If $i \in S_{\texttt{act}}(t)$ then

$$L^*(t-1) = L_i(t-1) \leq L_i(t) \leq L^*(t)$$

Else i is de-activated in round t, so

$$L^*(t-1) = L_i(t-1) \leq L_i(t) \leq U_i(t) < L^*(t).$$

(C2). Let us use induction on t. For $t = 0$, $L^*(t, \cdot) = 0$. Suppose the claim holds for $t-1$, and let us it for t. Suppose, for the sake of contradiction, that $L^*(t, b) > L^*(t, b^+)$. Let $j \in S_{\mathtt{act}}(t, b)$ be an agent such that $L_j(t, b) = L^*(t, b)$. Then by Eq. (17) it follows that $j \notin S_{\mathtt{act}}(t, b^+)$. Thus with bid vector b^+ agent j gets disqualified after some round $s < t$. Thus, using Part (C1),

$$U_j(s, b^+) < L^*(s, b^+) \leq L^*(t, b^+).$$

Now using Eq. (18) and Eq. (17) (for the right-most inequality), we get that

$$L^*(t, b) = L_j(t, b) \leq U_j(t, b) \leq U_j(s, b) \leq U_j(s, b^+).$$

Thus, $L^*(t, b) \leq L^*(t, b^+)$, the desired contradiction.

(C3). Use induction on t. The claim trivially holds for $t = 1$. Assuming the claim holds for some t we prove it holds for $t+1$. Fix agent $j \in S_{\mathtt{act}}(t, b^+) \setminus \{i\}$. Then $j \in S_{\mathtt{act}}(t, b)$ by the induction hypothesis, so by Eq. (17) $U_j(t, b) = U_j(t, b^+)$. Moreover, $L^*(t, b) \leq L^*(t, b^+)$ by Part (C2). Thus, agent j is de-activated with bid vector b only if it is de-activated with bid vector b^+.

(C4). Using Part (C3) and property Eq. (17), we can show that in each round t we have

$$\text{either } L^*(t, b^+) = L^*(t, b) \text{ or } L^*(t, b^+) = L_i(t, b^+).$$

Now let us use induction on t. Suppose $i \in S_{\mathtt{act}}(t+1, b)$. Then $U_i(t, b) \geq L^*(t, b)$, and we need to show $U_i(t, b^+) \geq L^*(t, b^+)$. Note that $i \in S_{\mathtt{act}}(t, b)$, so $i \in S_{\mathtt{act}}(t, b^+)$ by the induction hypothesis, and so $U_i(t, b^+) \geq U_i(t, b)$ by property Eq. (17). Thus, $U_i(t, b^+) \geq L^*(t, b)$. Also, $U_i(t, b^+) \geq L_i(t, b^+)$ by property Eq. (17). Therefore,

$$U_i(t, b^+) \geq \min(L^*(t, b),\, L_i(t, b^+)) \geq L^*(t, b^+).\square$$

Ex-post monotonicity follows easily from (C3-C4). Fix round t and let $q_i(t, b)$ be the probability that with bid vector b, agent i receives an impression in this round.

Claim 5.13. *For each round i, $q_i(t, b) \leq q_i(t, b^+)$.*

Proof. If $q_i(t, b) > 0$ then agent i is active with bid vector b, and hence (using (C4)) with b^+. If agent i is the designated agent in round t, then $q_i(t, b) = q_i(t, b^+) = 1$. Else, $q_i(t, \cdot) = 1/|S_{\mathtt{act}}(t, \cdot)|$, and so $q_i(t, b) \leq q_i(t, b^+)$ since by (C3) $|S_{\mathtt{act}}(t, b^+)| \leq |S_{\mathtt{act}}(t, b)|$. $\square$

Regret. The regret analysis is relatively standard, following the ideas in [5]. For simplicity assume that $b_{\text{max}} = 1$. Fix a bid vector b. For each agent i, let $c_i(t)$ and $n_i(t)$ be, respectively, the number of clicks and impressions in rounds $s \leq t$ when it is the designated agent. Let $r_i(t) = \sqrt{8 \log(T)/n_i(t)}$. Then $|\mu_i - c_i/n_i| \leq r_i(t)$ with probability at least $1 - T^{-3}$. In what follows, let us assume that this event holds. Then it easily follows from the specs of `NewCB` that

$$\begin{cases} L_i(t, b) \leq b_i\, \mu_i \leq U_i(t, b) \\ U_i(t, b) - L_i(t, b) \leq 4\, b_i\, \mu_i. \end{cases}$$

Let $\Delta_i = (\max_j b_j\, \mu_j) - b_i\, \mu_i$ be the suboptimality of i. Then each agent i with $\Delta_i > 0$ gets de-activated after $O(k\, \Delta_i^{-2} \log T)$ rounds, hence the bounds on regret.

5.4 The power of randomization

A by-product of Theorem 5.11 is a separation between the power of deterministic and randomized mechanisms, in terms of regret for MAB mechanisms that are ex-post truthful and ex-post normalized. The lower bound for deterministic mechanisms is from [8].

One challenge here is to ensure that the upper and lower bounds talk about exactly the same problem; as stated, Theorem 5.11 and the main lower bound result from [8] do not. To bypass this problem, we focus on the case of two agents, and use a more general version of the lower bound: Theorem C.1 in the full version of [8]. Further, to match [8] we extend the mechanism from Theorem 5.11 to a setting in which b_{max} is not known a priori.

We formulate the separation theorem as follows. Denote $R(T, b_{\text{max}}) \triangleq \max R(T; b; \mu)$, where the maximum is taken over all CTR vectors μ and all bid vectors b such that $b_i \leq b_{\text{max}}$ for all i.

Theorem 5.14. *Consider the stochastic MAB mechanism design problem with two agents. Assume b_{max} is not known a priori to the mechanism. Suppose $\mathcal{M}$ is an MAB mechanism that is (i) ex-post truthful and ex-post normalized, and (ii) has regret $R(T, b_{max}) = \tilde{O}(b_{max}\, T^{\gamma})$ for some γ and any b_{max}. Then:*

(a) [8] If $\mathcal{M}$ is deterministic then $\gamma \geq \frac{2}{3}$.

(b) There exists such randomized *$\mathcal{M}$ with $\gamma = \frac{1}{2}$.*

Proof of part (b). Let $\mathcal{A}'$ be the ex-post monotone MAB allocation rule in Theorem 5.11, for $b_{\text{max}} = 1$. Define an MAB allocation rule $\mathcal{A}$ as a rule that inputs the bid vector b and passes the modified bid vector $b' = b/(\max_i b_i)$ to $\mathcal{A}'$. We claim that $\mathcal{A}$ is ex-post monotone, too. Indeed, w.l.o.g. assume $b_1 > b_2$. If b_2 increases (to a value $\leq b_1$), then b'_2 increases while b'_1 stays the same. Thus, the total click reward of agent 2 increases. If b_1 increases then b'_2 decreases while b'_1 stays the same, so the total click reward of agent 2 does not increase, which implies that the total click reward of agent 1 does not decrease. Claim proved. Now part (b) follows from Theorem 5.1 (with $\mu = \frac{1}{T}$). □

6 Open Questions

This paper gives rise to a number of open questions, which fall into three directions. The first direction concerns the "quality" of our general transformation: is it the best possible? One concrete way to phrase this is whether it obtains the optimal trade-off between the loss in welfare and the maximal rebate size? For positive types, what if no rebates (i.e., positive transfers) are allowed? The second direction concerns the power of randomization for mechanism design. We have a separation result for ex-post truthful MAB mechanisms. Can one obtain similar separation results for other single-parameter domains? Third, Theorem 5.1 opens up the "monotone MAB allocation design problem", valid for any MAB setting in the literature. Perhaps the most interesting question here is whether one can match the optimal regret for the adversarial MAB setting.

Acknowledgments

We are indebted to Tim Roughgarden for suggesting that for positive types, an improved bound on the social welfare is possible (see Section 3.5). We would like to acknowledge that the preliminary form of a generalization of our "generic transformation" to negative bids, and (the preliminary form of) the applications to offline mechanism design, have been derived jointly with Jason Hartline.

References

[1] A. Archer, C. Papadimitriou, K. Talwar, and É. Tardos. An approximate truthful mechanism for combinatorial auctions with single parameter agents. *Internet Mathematics*, 1:129–150, 2004. Extended abstract in *SODA 2003*.

[2] A. Archer and É. Tardos. Truthful mechanisms for one-parameter agents. In *42nd IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 482–491, 2001.

[3] S. Athey and I. Segal. An efficient dynamic mechanism. Working Paper, Mar. 2007.

[4] J.-Y. Audibert and S. Bubeck. Minimax policies for adversarial and stochastic bandits. In *22nd Conf. on Learning Theory (COLT)*, 2009.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002. Preliminary version in *15th ICML*, 1998.

[6] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002. Preliminary version in *36th IEEE FOCS*, 1995.

[7] M. Babaioff, L. Blumrosen, M. Naor, and M. Schapira. Informational overhead of incentive compatibility. In *9th ACM Conf. on Electronic Commerce (EC)*, pages 88–97, 2008.

[8] M. Babaioff, Y. Sharma, and A. Slivkins. Characterizing truthful multi-armed bandit mechanisms. In *10th ACM Conf. on Electronic Commerce (EC)*, pages 79–88, 2009. Full version available on `arxiv.org`.

[9] D. Bergemann and J. Välimäki. Efficient dynamic auctions. Cowles Foundation Discussion Paper CFDP 1584, Oct. 2006.

[10] N. Devanur and S. M. Kakade. The price of truthfulness for pay-per-click auctions. In *10th ACM Conf. on Electronic Commerce (EC)*, pages 99–106, 2009.

[11] S. Dobzinski and S. Dughmi. On the power of randomization in algorithmic mechanism design. In *50th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2009.

[12] R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. In *21st Conf. on Learning Theory (COLT)*, pages 425–436, 2008.

[13] T. Lai and H. Robbins. Asymptotically efficient Adaptive Allocation Rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

[14] R. B. Myerson. Optimal Auction Design. *Mathematics of Operations Research*, 6:58–73, 1981.

[15] N. Nisan and A. Ronen. Algorithmic Mechanism Design. *Games and Economic Behavior*, 35(1-2):166–196, 2001.

A Non-recursive self-resampling

In this section we prove Proposition 3.6 which asserts that $\textsc{Recursive}_\mu$ and $\textsc{OneShot}_\mu$ (as defined in Algorithm 1 and Algorithm 2, respectively) generate the same output distribution.

In what follows, we will omit the subscript i from the description of the procedures. That is, a self-resampling procedure inputs a scalar bid b and a random seed w, and outputs two numbers (x, y). For convenience, let us restate Proposition 3.6.

Proposition A.1. *$\textsc{Recursive}_\mu$ and $\textsc{OneShot}_\mu$ generate the same output distribution: for any bid $b \in [0, \infty)$, the joint distribution of the pair $(x, y) = (x(b; w), y(b; w))$ is the same for both procedures.*

The proof of the proposition uses the following well-known fact about exponential distributions which we state without a proof:

Lemma A.2. *Let $X_1, \ldots, X_n$ be drawn independently, uniformly at random from $(0, 1)$. Let $F(x, p) = -\ln(1 - x)/p$, and let $p_1, \ldots, p_n$ be positive numbers that sum up to 1. Then each $Y_i = F(X_i, p_i)$ is an exponentially distributed random variable with rate p_i. Moreover, $\Pr[Y_i = \min_j Y_j] = p_i$.*

Let us restate $\textsc{Recursive}_\mu$ and $\textsc{OneShot}_\mu$ so that the random seed is explicit. Both procedures will use a random seed w consisting of two infinite sequences $(\beta_1, \beta_2, \ldots)$ and $(\gamma_1, \gamma_2, \ldots)$ such that the β_j are i.i.d. Bernoulli random variables with $\mathbb{E}(\beta_j) = 1-\mu$, and the γ_j are i.i.d. uniform random variables in $[0, 1]$. Then the two procedures can be restated as follows: two procedures.

$\textsc{Recursive}_\mu(b; w)$**:** If $\beta_1 = 1$, output $x(b; w) = y(b; w) = b$. Otherwise, let $b' = b \cdot \gamma_1$, and let w' denote the *shifted random seed* consisting of the sequences $(\beta_2, \beta_3, \ldots)$ and $(\gamma_2, \gamma_3, \ldots)$. Output $x(b; w) = x(b'; w')$ and $y(b; w) = b'$. Note that the recursive procedure for calculating x terminates as long as $\beta_j = 1$ for at least one value of j, an event that has probability 1.

$\textsc{OneShot}_\mu(b; w)$**:** If $\beta_1 = 1$, output $x(b; w) = y(b; w) = b$. Otherwise, output $x(b; w) = b \cdot \gamma_1^{1/(1-\mu)}$ and $y(b; w) = b \cdot \max\{\gamma_1^{1/(1-\mu)}, \gamma_2^{1/\mu}\}$.

It will be useful to consider the joint distribution of $(l_x, l_y) = (\log(b/x), \log(b/y))$ when $(x, y) = (x(b; w), y(b; w))$ is sampled using either $\textsc{Recursive}_\mu$ or $\textsc{OneShot}_\mu$. From the descriptions of the two algorithms, and the fact that $\log(1/x)$ is exponentially distributed with mean 1 when x is uniformly distributed on $[0, 1]$, we derive the following two sampling procedures for (l_x, l_y).

Procedure 1: With probability $1 - \mu$ output $l_x = l_y = 0$. Otherwise, repeatedly sample an independent exponential random variable v_i with mean 1 and Bernoulli random variable w_i with mean $1 - \mu$ until the first t such that $w_t = 1$. Output $l_x = \sum_{i=1}^{t} v_i$ and $l_y = v_1$.

Procedure 2: With probability $1 - \mu$ output $l_x = l_y = 0$. Otherwise, sample two independent exponential random variables u_1, u_2 with means $1/(1 - \mu)$ and $1/\mu$ respectively. Output $l_x = u_1$ and $l_y = \min(u_1, u_2)$.

Our goal is to prove that Procedures 1 and 2 generate the same joint distribution of (l_x, l_y); clearly this reduces to proving that the conditional distribution of (l_x, l_y) given that $(l_x, l_y) \neq (0, 0)$ is the same in both cases. To this end, let $v_1, v_2, \ldots$ denote an infinite sequence of independent exponentially distributed random variables, each with mean 1, and let $w_1, w_2, \ldots$ denote an infinite sequence of independent Bernoulli random variables, each with mean $1 - \mu$. Consider the sequence of partial sums $s_1, s_2, \ldots$ given by $s_i := \sum_{j=1}^{i} v_i$. Let $r_1 = \min\{s_i : w_i = 1\}$ and $r_2 = \min\{s_i : w_i = 0\}$. Conditional on the event that $(l_x, l_y) \neq (0, 0)$, Procedure 1 outputs (l_x, l_y) with the same distribution as $(r_1, \min\{r_1, r_2\})$. Recall that Procedure 2 outputs $(l_x, l_y) = (u_1, \min\{u_1, u_2\})$. Thus, to finish the proof, it suffices to prove that (r_1, r_2) have the same joint distribution as (u_1, u_2). We prove this fact using a sequence of three observations.

1. First, $\min\{r_1, r_2\}$ has the same distribution as $\min\{u_1, u_2\}$: both are exponentially distributed with mean 1. For $\min\{r_1, r_2\}$ this is obvious from the definition, for $\min\{u_1, u_2\}$ it follows from Lemma A.2.

2. Second, conditioning on the value of $\min\{r_1, r_2\}$ and the value of $\min\{u_1, u_2\}$, the probability that $\min\{r_1, r_2\} = r_1$ and the probability that $\min\{u_1, u_2\} = u_1$ are both equal to $1 - \mu$. Once again, for r_1, r_2 this fact is obvious from the definitions, whereas for u_1, u_2 it is a consequence of Lemma A.2 combined with the fact that an exponential random variable u is "memoryless": the conditional distribution of $u - q$ given that $u > q$ is the same as the unconditional distribution of u. (In more detail, the memorylessness, together with Lemma A.2, implies that for all q, the conditional probability that $\min\{u_1, u_2\} = u_1$ given $\min\{u_1, u_2\} > q$ is always equal to $1 - \mu$. Since this conditional probability is independent of q, it must remain the same when we condition on $\min\{u_1, u_2\} = q$ rather than $\min\{u_1, u_2\} > q$.)

3. Third, conditioning on the event that $\min\{r_1, r_2\} = r_i$ for $i \in \{1, 2\}$, the random variable $\max\{r_1, r_2\} - \min\{r_1, r_2\}$ is exponentially distributed with mean m_i, where $m_i = 1/\mu$ if $i = 1$ and $m_i = 1/(1-\mu)$ if $i = 2$. Moreover, the same fact applies with (u_1, u_2) in place of (r_1, r_2). In this case, the claim is obvious for (u_1, u_2) because each of them is memoryless and independent of the other. To prove the claim for (r_1, r_2), note from the definition of r_1 and r_2 that the conditional distribution of $\max\{r_1, r_2\} - \min\{r_1, r_2\}$ is equal to the unconditional distribution of $V := s_k - s_1 = \sum_{i=2}^{k} v_i$ where the v_i are independent exponential random variables with mean 1, and $k - 1$ is geometrically distributed with mean m_i, independent of the sequence (v_i). This sum has expected value m_i, so it remains to show that it is exponentially distributed, i.e. memoryless. If $V > q$ then there is exactly one value of j such that $s_j - s_1 \leq q < s_{j+1} - s_1$, and $k > j$. Conditioning on the event $V > q$ and on the value of j, we observe the following: the random variables $s_{j+1} - q, v_{j+2}, v_{j+3}, \ldots$ are independent and exponentially distributed with mean 1, and the random variable $k - j$ is geometrically distributed with mean m_i. Thus, the conditional distribution of $V - q = (s_{j+1} - q) + \sum_{j+1<\ell\leq k} v_\ell$ is exactly the same as the unconditional distribution of V. Now removing the conditioning on j, we see that the conditional distribution of $V - q$ given the event $V > q$ is the same as the unconditional distribution of V, i.e. V is memoryless, as desired.